

Appl. No. 10/682,387

Response to 1st Office Action dated 07/17/2006

Reply to 1st Office Action of 04/17/2006

IN THE CLAIMS:

This listing of claims will replace all prior versions, and listing of claims, in the Application.

Listing of claims:

1. (Currently amended) A method of enhancing priority boosting of scheduled threads comprising the steps of:

determining by a second thread being executed on a second CPU whether to wait for a lock on a shared resource held by a first thread that is scheduled to be executed by a first CPU, the second thread having a higher priority than the first thread;

boosting the priority of the first thread by passing the higher priority of the second thread to the first thread if the second thread has to wait for the lock on the shared thread; and

enhancing the priority boosting of the first thread by rescheduling the first thread to run on the second CPU.

2. (Original) The method of Claim 1 wherein when the priority boosting of the first thread is enhanced, the first thread becomes the next thread to be dispatched for execution.
3. (Original) The method of Claim 2 wherein after the priority boosting of the first thread has been enhanced, the second thread relinquishes the second CPU by going to sleep awaiting the release of the lock by the first thread.

AUS920030439US1

Appl. No. 10/682,387

Response to 1st Office Action dated 07/17/2006

Reply to 1st Office Action of 04/17/2006

4. (Original) The method of Claim 3 wherein after the first thread has released the lock, the second thread is awakened and rescheduled for execution by the second CPU.
 5. (Original) The method of Claim 4 wherein after being rescheduled for execution by the second CPU, the second thread is likely the next thread to be dispatched for execution.
 6. (Currently amended) A computer program product on a computer readable medium for enhancing priority boosting of scheduled threads comprising:

code means for determining by a second thread being executed on a second CPU whether to wait for a lock on a shared resource held by a first thread that is scheduled to be executed by a first CPU, the second thread having a higher priority than the first thread;

code means for boosting the priority of the first thread by passing the higher priority of the second thread to the first thread if the second thread has to wait for the lock on the shared thread; and

code means for enhancing the priority boosting of the first thread by rescheduling the first thread to run on the second CPU.
 7. (Original) The computer program product of Claim 6 wherein when the priority boosting of the first thread is enhanced, the first thread becomes the next thread to be dispatched for execution.
 8. (Original) The computer program product of Claim 7 wherein after the priority boosting of the first thread has been enhanced, the second thread
- AUS920030439US1

Appl. No. 10/682,387

Response to 1st Office Action dated 07/17/2006

Reply to 1st Office Action of 04/17/2006

relinquishes the second CPU by going to sleep awaiting the release of the lock by the first thread.

9. (Original) The computer program product of Claim 8 wherein after the first thread has released the lock, the second thread is awakened and rescheduled for execution by the second CPU.

10. (Original) The computer program product of Claim 9 wherein after being rescheduled for execution by the second CPU, the second thread is likely the next thread to be dispatched for execution.

11. (Currently amended) An apparatus for enhancing priority boosting of scheduled threads comprising:

means for determining by a second thread being executed on a second CPU whether to wait for a lock on a shared resource held by a first thread that is scheduled to be executed by a first CPU, the second thread having a higher priority than the first thread;

means for boosting the priority of the first thread by passing the higher priority of the second thread to the first thread if the second thread has to wait for the lock on the shared thread; and

means for enhancing the priority boosting of the first thread by rescheduling the first thread to run on the second CPU.

12. (Original) The apparatus of Claim 11 wherein when the priority boosting of the first thread is enhanced, the first thread becomes the next thread to be dispatched for execution.

AUS920030439US1

Appl. No. 10/682,387

Response to 1st Office Action dated 07/17/2006

Reply to 1st Office Action of 04/17/2006

13. (Original) The apparatus of Claim 12 wherein after the priority boosting of the first thread has been enhanced, the second thread relinquishes the second CPU by going to sleep awaiting the release of the lock by the first thread.
14. (Original) The apparatus of Claim 13 wherein after the first thread has released the lock, the second thread is awakened and rescheduled for execution by the second CPU.
15. (Original) The apparatus of Claim 14 wherein after being rescheduled for execution by the second CPU, the second thread is likely the next thread to be dispatched for execution.
16. (Currently amended) A system for enhancing priority boosting of scheduled threads comprising:

at least one storage device for storing code data; and

at least two CPUs, one of the at least two CPUs for processing the code data to determine by a second thread being executed on the at least one CPU whether to wait for a lock on a shared resource held by a first thread that is scheduled to be executed by the other CPU, the second thread having a higher priority than the first thread, to boost the priority of the first thread by passing the higher priority of the second thread to the first thread if the second thread has to wait for the lock on the shared thread, and to enhance the priority boosting of the first thread by rescheduling the first thread to run on the second CPU.

AUS920030439US1

Appl. No. 10/682,387

Response to 1st Office Action dated 07/17/2006

Reply to 1st Office Action of 04/17/2006

17. (Original) The system of Claim 16 wherein when the priority boosting of the first thread is enhanced, the first thread becomes the next thread to be dispatched for execution.
18. (Original) The system of Claim 17 wherein after the priority boosting of the first thread has been enhanced, the second thread relinquishes the second CPU by going to sleep awaiting the release of the lock by the first thread.
19. (Original) The system of Claim 18 wherein after the first thread has released the lock, the second thread is awakened and rescheduled for execution by the second CPU.
20. (Original) The system of Claim 19 wherein after being rescheduled for execution by the second CPU, the second thread is likely the next thread to be dispatched for execution.

AUS920030439US1